RESEARCHER DEVELOPMENT

Clustering
(unsupervised learning)

John Joseph Valletta

University of Exeter, Penryn Campus, UK

1st-2nd June 2015

www.exeter.ac.uk/as/rdp/

Overview

- What is clustering?
- Major types of clustering methods
- k-means clustering
- Agglomerative hierarchical clustering
- Gaussian mixture models
- How do we determine the correct number of clusters?

Overview

- What is clustering?
- Major types of clustering methods
- k-means clustering
- Agglomerative hierarchical clustering
- Gaussian mixture models
- How do we determine the correct number of clusters?

Overview

- What is clustering?
- Major types of clustering methods
- k-means clustering
- Agglomerative hierarchical clustering
- Gaussian mixture models
- How do we determine the correct number of clusters?

Overview

- What is clustering?
- Major types of clustering methods
- k-means clustering
- Agglomerative hierarchical clustering
- Gaussian mixture models
- How do we determine the correct number of clusters?

Overview

- What is clustering?
- Major types of clustering methods
- k-means clustering
- Agglomerative hierarchical clustering
- Gaussian mixture models
- How do we determine the correct number of clusters?

Overview

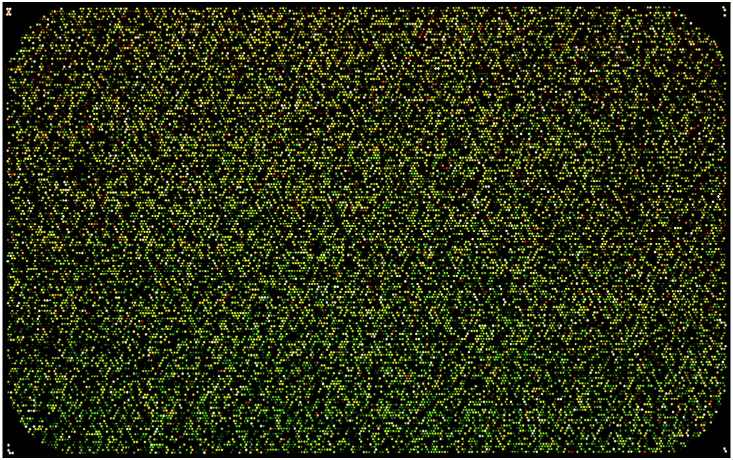
- What is clustering?
- Major types of clustering methods
- k-means clustering
- Agglomerative hierarchical clustering
- Gaussian mixture models
- How do we determine the correct number of clusters?

Overview

- What is clustering?
- Major types of clustering methods
- k-means clustering
- Agglomerative hierarchical clustering
- Gaussian mixture models
- How do we determine the correct number of clusters?

What's the problem?

Well, it's driven by your data, e.g which of these genes are co-regulated?



We want to find some underlying structure in the data → **Clustering**

What's the problem?

Gene expression:	discovering co-regulated genes
Biological systematics:	finding organisms sharing similar attributes
Computer vision:	segmenting a digital image for object recognition
Epidemiology:	identifying geographical clusters of diseases
Medical imaging:	differentiating between tissues
Mathematical chemistry:	grouping compounds by topological indices

Clustering is particularly useful in applications where labelling the data is very time consuming/expensive

What is clustering?

Formal definition

Identifying homogeneous and well separated groups of data points (features) by some similarity measure

What is clustering?

Formal definition

Identifying homogeneous and well separated groups of data points (features) by some similarity measure

Informal definition

The process of stereotyping your data
e.g *these* are round(ish) faces, *these* are short(ish) people

What is clustering?

Formal definition

Identifying homogeneous and well separated groups of data points (features) by some similarity measure

Informal definition

The process of stereotyping your data
e.g *these* are round(ish) faces, *these* are short(ish) people

But how many groups?

An unsolved problem. Issue lies in the subjectivity of the word **similar** and its mathematical definition

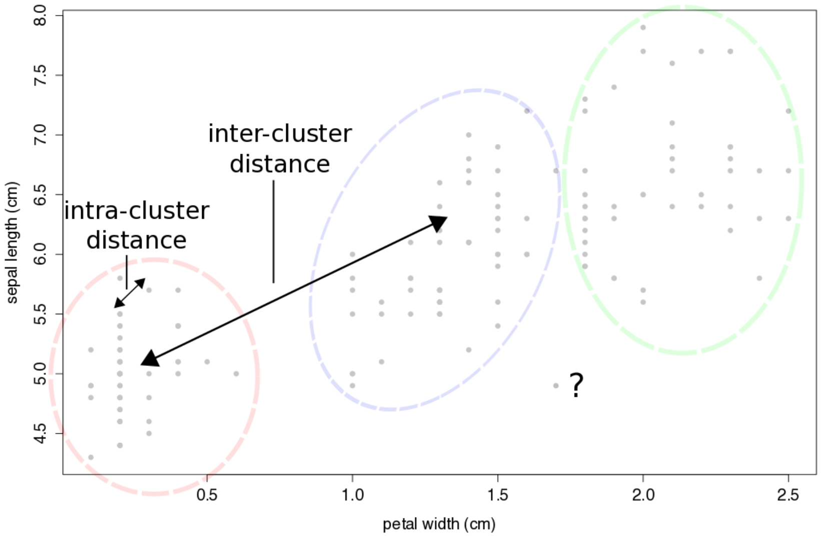
Are they similar?

Are they similar?

What are we after?

Motivation: How is the data structured? Any outliers?

Goals: *High* intra-cluster similarity and *low* inter-cluster similarity



Major types of clustering methods

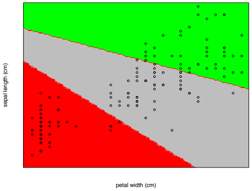
Partitional: The data (feature) space is partitioned into k regions

Hierarchical: Iteratively merging small clusters into larger ones (*agglomerative*) or breaking large clusters into smaller ones (*divisive*)

Distribution-based: Fit k multivariate statistical distributions

Major types of clustering methods

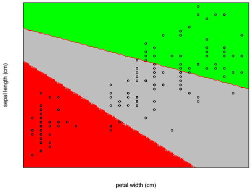
Partitional: The data (feature) space is partitioned into k regions

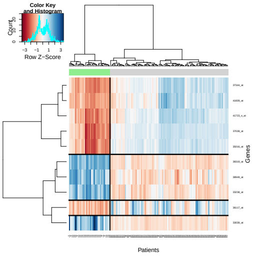


Hierarchical: Iteratively merging small clusters into larger ones (*agglomerative*) or breaking large clusters into smaller ones (*divisive*)

Distribution-based: Fit k multivariate statistical distributions

Major types of clustering methods





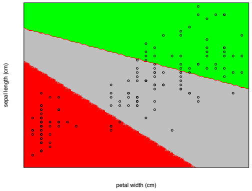
Partitional: The data (feature) space is partitioned into k regions

Hierarchical: Iteratively merging small clusters into larger ones (*agglomerative*) or breaking large clusters into smaller ones (*divisive*)

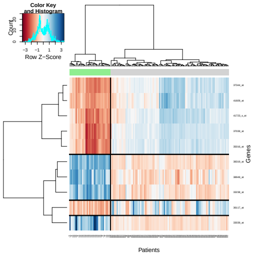
Distribution-based: Fit k multivariate statistical distributions

Major types of clustering methods

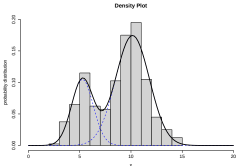
Partitional: The data (feature) space is partitioned into k regions



Hierarchical: Iteratively merging small clusters into larger ones (*agglomerative*) or breaking large clusters into smaller ones (*divisive*)

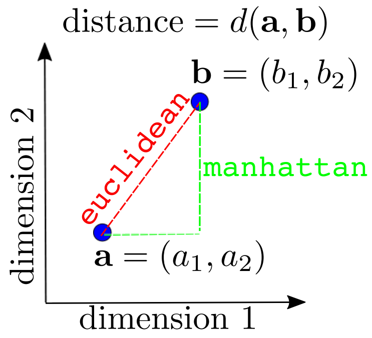


Distribution-based: Fit k multivariate statistical distributions



Similarity measures

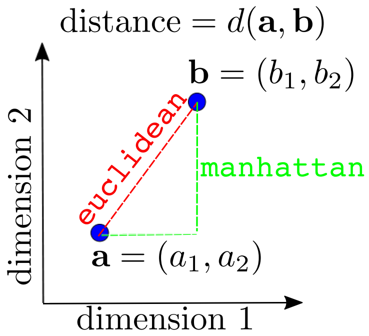
- A *distance* metric quantifies how *close* two data points are
- Several ways to define this distance which has a direct impact on the clustering result

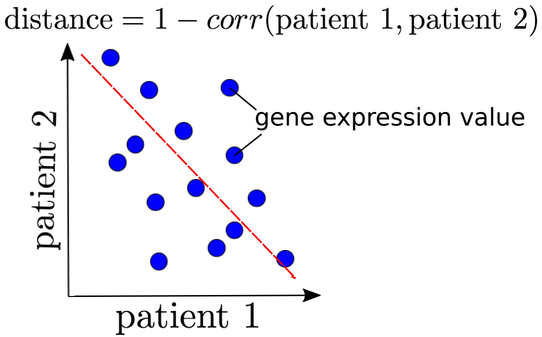


$\text{distance} = d(\mathbf{a}, \mathbf{b})$
$\mathbf{b} = (b_1, b_2)$
euclidean
manhattan
$\mathbf{a} = (a_1, a_2)$
dimension 2
dimension 1

Similarity measures

- A *distance* metric quantifies how *close* two data points are
- Several ways to define this distance which has a direct impact on the clustering result


$\text{distance} = d(\mathbf{a}, \mathbf{b})$
$\mathbf{b} = (b_1, b_2)$
euclidean
manhattan
$\mathbf{a} = (a_1, a_2)$
dimension 2
dimension 1


$\text{distance} = 1 - corr(\text{patient 1}, \text{patient 2})$
gene expression value
patient 2
patient 1

k-means clustering

```
fit <- kmeans(x, centers)
# x - numeric matrix of data
# centers - no. of clusters k
```

1. Select k centroids at random
2. Calculate distance between centroids and each data point
3. Assign each data point to the closest centroid
4. Compute new centroids; the average of all data points in that cluster
5. Repeat steps 2 to 4 until data points remain in the same cluster or some maximum number of iterations reached

k-means clustering

```r
fit <- kmeans(x, centers)
# x - numeric matrix of data
# centers - no. of clusters k
```

1. Select k centroids at random
2. Calculate distance between centroids and each data point
3. Assign each data point to the closest centroid
4. Compute new centroids; the average of all data points in that cluster
5. Repeat steps 2 to 4 until data points remain in the same cluster or some maximum number of iterations reached

k-means clustering

```
fit <- kmeans(x, centers)
# x - numeric matrix of data
# centers - no. of clusters k
```

1. Select k centroids at random
2. Calculate distance between centroids and each data point
3. Assign each data point to the closest centroid
4. Compute new centroids; the average of all data points in that cluster
5. Repeat steps 2 to 4 until data points remain in the same cluster or some maximum number of iterations reached

k-means clustering

```
fit <- kmeans(x, centers)
# x - numeric matrix of data
# centers - no. of clusters k
```

1. Select k centroids at random
2. Calculate distance between centroids and each data point
3. Assign each data point to the closest centroid
4. Compute new centroids; the average of all data points in that cluster
5. Repeat steps 2 to 4 until data points remain in the same cluster or some maximum number of iterations reached

k-means clustering

```
fit <- kmeans(x, centers)
# x - numeric matrix of data
# centers - no. of clusters k
```

1. Select k centroids at random
2. Calculate distance between centroids and each data point
3. Assign each data point to the closest centroid
4. Compute new centroids; the average of all data points in that cluster
5. Repeat steps 2 to 4 until data points remain in the same cluster or some maximum number of iterations reached

k-means clustering

```r
fit <- kmeans(x, centers)
# x - numeric matrix of data
# centers - no. of clusters k
```

1. Select k centroids at random
2. Calculate distance between centroids and each data point
3. Assign each data point to the closest centroid
4. Compute new centroids; the average of all data points in that cluster
5. Repeat steps 2 to 4 until data points remain in the same cluster or some maximum number of iterations reached

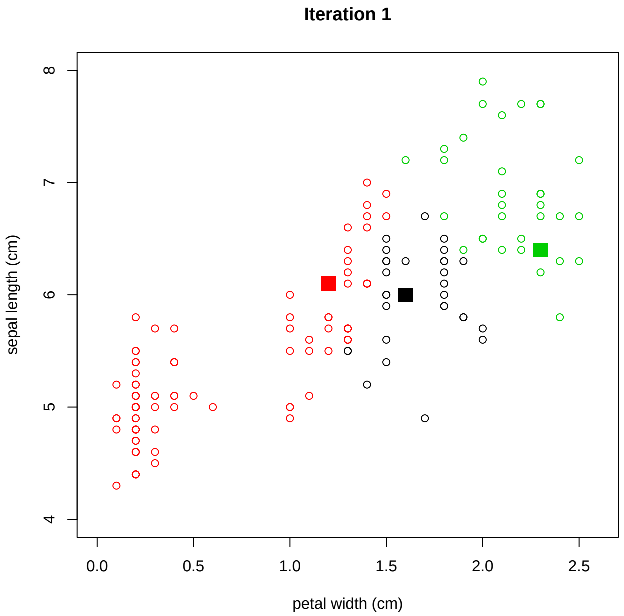
k-means clustering

```
fit <- kmeans(x, centers)
# x - numeric matrix of data
# centers - no. of clusters k
```

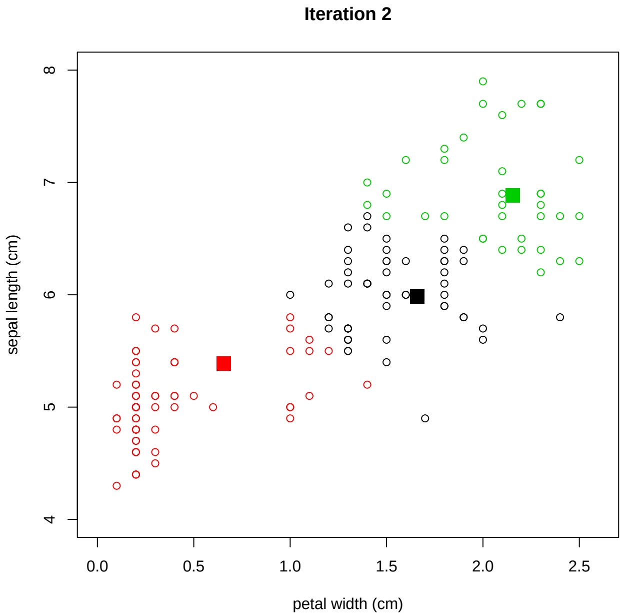
1. Select k centroids at random
2. Calculate distance between centroids and each data point
3. Assign each data point to the closest centroid
4. Compute new centroids; the average of all data points in that cluster
5. Repeat steps 2 to 4 until data points remain in the same cluster or some maximum number of iterations reached

Note: k-means clustering should *only* be used with continuous data

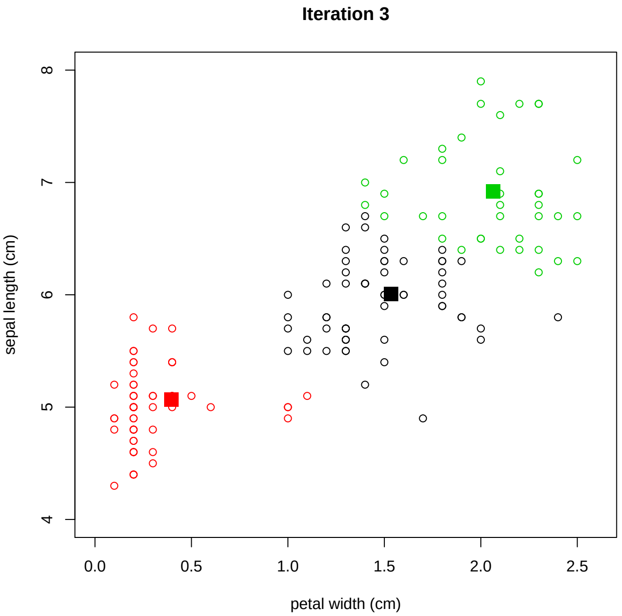
k-means clustering



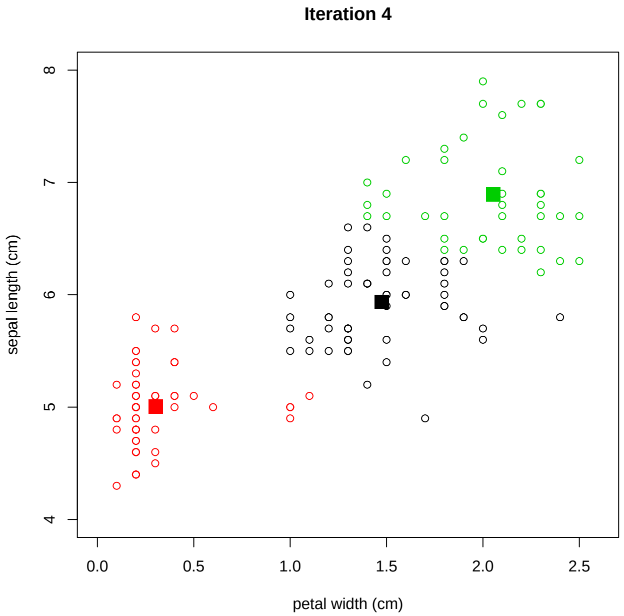
k-means clustering



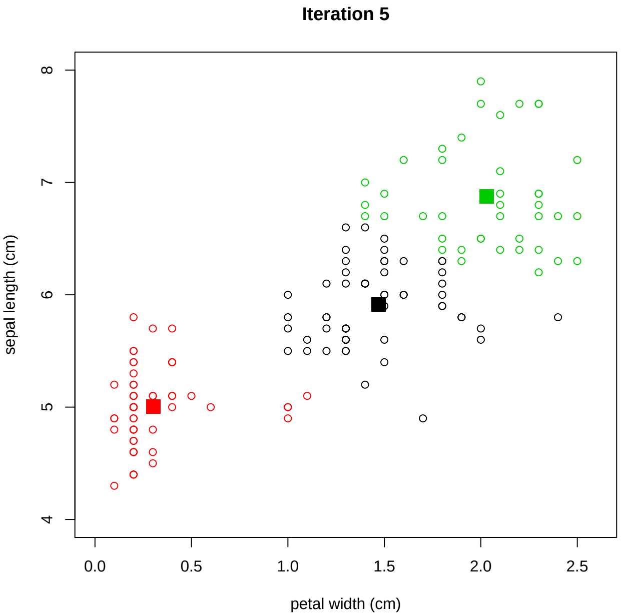
k-means clustering



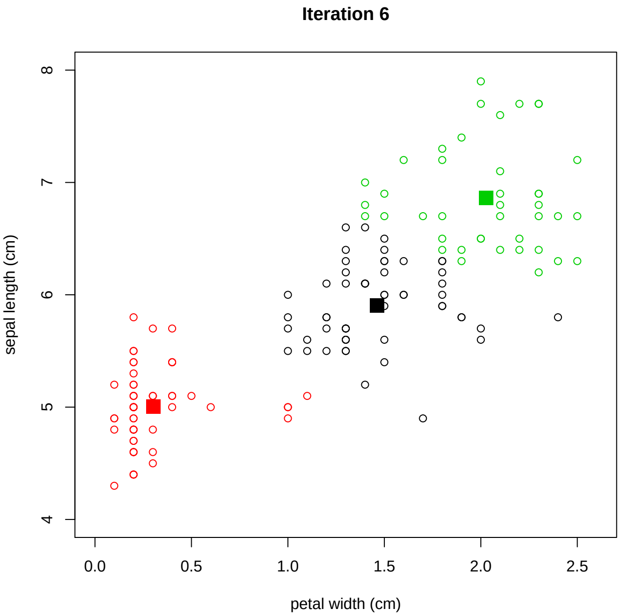
k-means clustering



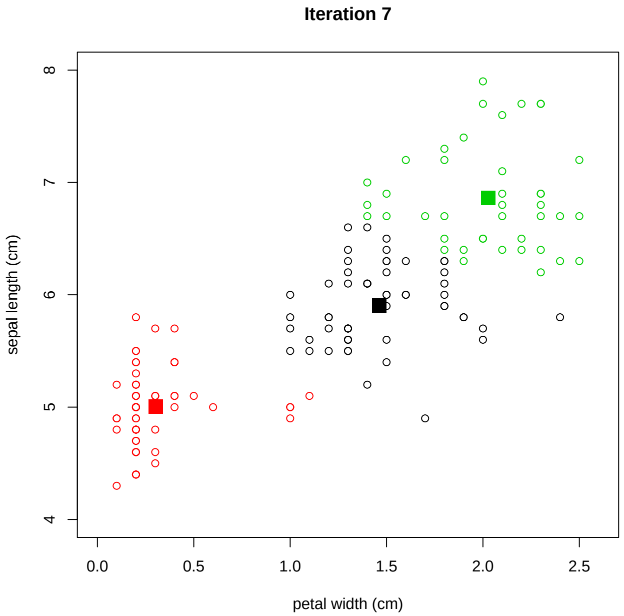
k-means clustering



k-means clustering



k-means clustering



k-means clustering

Pros

- Simple and intuitive
- Computationally inexpensive/fast

Cons

- What is k?
- Only applicable to continuous data where a mean is defined
- No guarantee of a global optimum solution

Agglomerative hierarchical clustering

```
d <- dist(as.matrix(data), method)
# data - data frame
# method - distance method e.g "euclidean" or "manhattan"
fit <- hclust(d, method)
# method - linkage function e.g "complete" or "single"
```

1. Assign each data point as its own cluster
2. Compute distance between each cluster
3. Merge the closest pair into a single cluster
4. Repeat 2 to 3 until you're left with one cluster

Agglomerative hierarchical clustering

```r
d <- dist(as.matrix(data), method)
# data - data frame
# method - distance method e.g "euclidean" or "manhattan"
fit <- hclust(d, method)
# method - linkage function e.g "complete" or "single"
```

1. Assign each data point as its own cluster
2. Compute distance between each cluster
3. Merge the closest pair into a single cluster
4. Repeat 2 to 3 until you're left with one cluster

Agglomerative hierarchical clustering

```r
d <- dist(as.matrix(data), method)
# data - data frame
# method - distance method e.g "euclidean" or "manhattan"
fit <- hclust(d, method)
# method - linkage function e.g "complete" or "single"
```

1. Assign each data point as its own cluster
2. Compute distance between each cluster
3. Merge the closest pair into a single cluster
4. Repeat 2 to 3 until you're left with one cluster

Agglomerative hierarchical clustering

```
d <- dist(as.matrix(data), method)
# data - data frame
# method - distance method e.g "euclidean" or "manhattan"
fit <- hclust(d, method)
# method - linkage function e.g "complete" or "single"
```

1. Assign each data point as its own cluster
2. Compute distance between each cluster
3. Merge the closest pair into a single cluster
4. Repeat 2 to 3 until you're left with one cluster

Agglomerative hierarchical clustering

```r
d <- dist(as.matrix(data), method)
# data - data frame
# method - distance method e.g "euclidean" or "manhattan"
fit <- hclust(d, method)
# method - linkage function e.g "complete" or "single"
```

1. Assign each data point as its own cluster
2. Compute distance between each cluster
3. Merge the closest pair into a single cluster
4. Repeat 2 to 3 until you're left with one cluster

Agglomerative hierarchical clustering

```
d <- dist(as.matrix(data), method)
# data - data frame
# method - distance method e.g "euclidean" or "manhattan"
fit <- hclust(d, method)
# method - linkage function e.g "complete" or "single"
```

1. Assign each data point as its own cluster
2. Compute distance between each cluster
3. Merge the closest pair into a single cluster
4. Repeat 2 to 3 until you're left with one cluster

Note: Step 3 is *key*, the distance method and linkage function dictate the final result

Hierarchical clustering: Link method

How do we compute the inter-cluster distance? The *linkage function*

Centroid: mean of data points (same as in k-means)

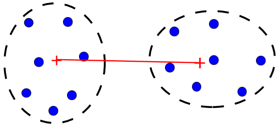
Single: distance between closest pair of points

Complete: distance between furthest pair of points

Average: mean pairwise distance between all points

Hierarchical clustering: Link method

How do we compute the inter-cluster distance? The *linkage function*



Centroid: mean of data points (same as in k-means)

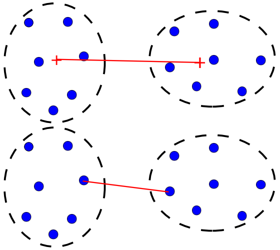
Single: distance between closest pair of points

Complete: distance between furthest pair of points

Average: mean pairwise distance between all points

Hierarchical clustering: Link method

How do we compute the inter-cluster distance? The *linkage function*



Centroid: mean of data points (same as in k-means)

Single: distance between closest pair of points

Complete: distance between furthest pair of points

Average: mean pairwise distance between all points

Hierarchical clustering: Link method

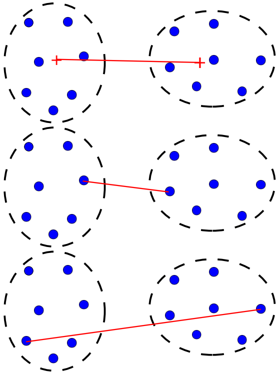
How do we compute the inter-cluster distance? The *linkage function*

Centroid: mean of data points (same as in k-means)

Single: distance between closest pair of points

Complete: distance between furthest pair of points

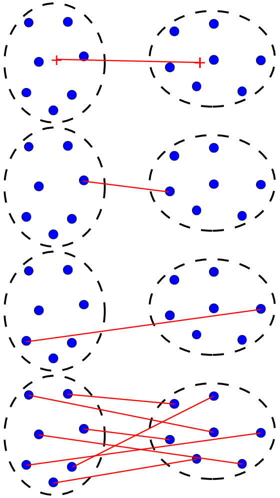
Average: mean pairwise distance between all points



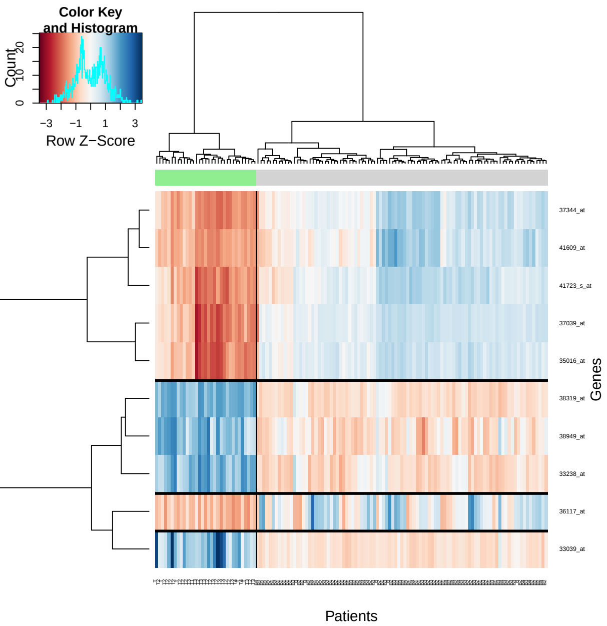
Hierarchical clustering: Link method

How do we compute the inter-cluster distance? The *linkage function*

Centroid:	mean of data points (same as in k-means)
Single:	distance between closest pair of points
Complete:	distance between furthest pair of points
Average:	mean pairwise distance between all points



Hierarchical clustering in gene expression studies



Hierarchical clustering

Pros

- No need to specify k
- Results can be visualised nicely irrespective of number of dimensions

Cons

- Can be computationally expensive
- Interpretation is subjective. Where should we draw the line (to separate clusters)?
- Choice of distance method and linkage function can significantly change the result

Gaussian mixture models

```r
library(mclust)
fit <- Mclust(data, G)
# data - data frame
# G - no. of Gaussians
```

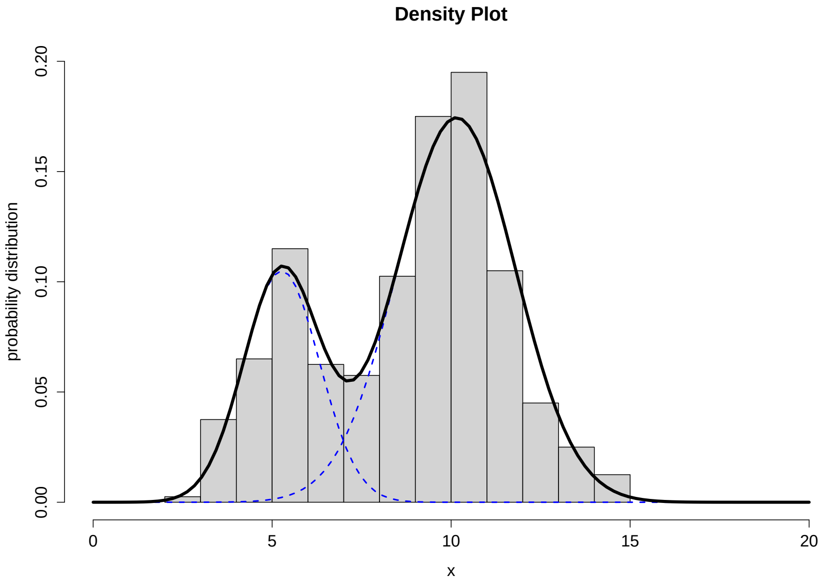
1. Fit k multivariate Gaussian distributions

The Expectation-Maximisation (EM) algorithm is used to estimate the parameters π_i (mixing coefficients), μ_i and σ_i

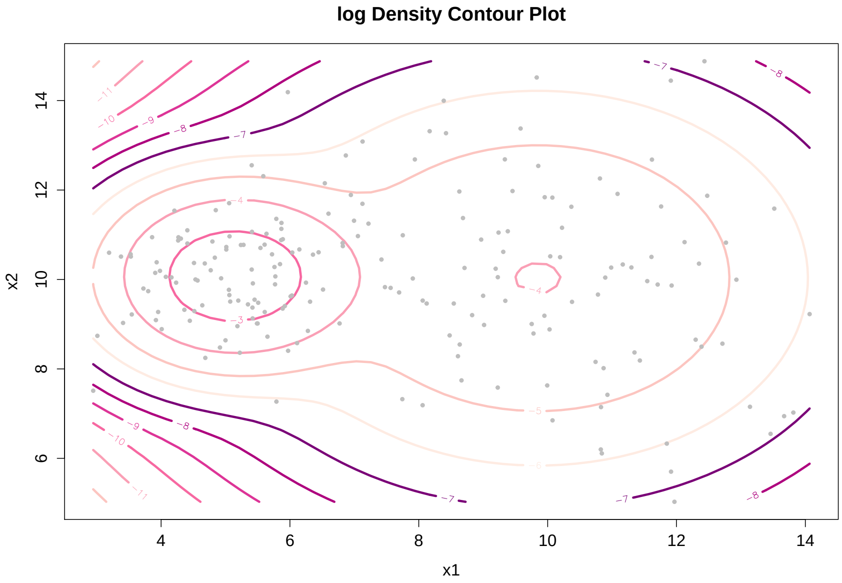
$$p(x) = \sum_{i=1}^{k} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i) \text{ and } \sum_{i=1}^{k} \pi_i = 1$$

Can be seen as a "soft" version of k-means because *every* point is part of *every* cluster but with varying levels of membership

Gaussian mixture models



Gaussian mixture models



Gaussian mixture models

Pros

- Intuitive interpretation
- Computationally inexpensive

Cons

- What is k?
- Strong assumption on the data (normality)
- No guarantee of a global optimum solution

How do we determine the correct number of clusters?

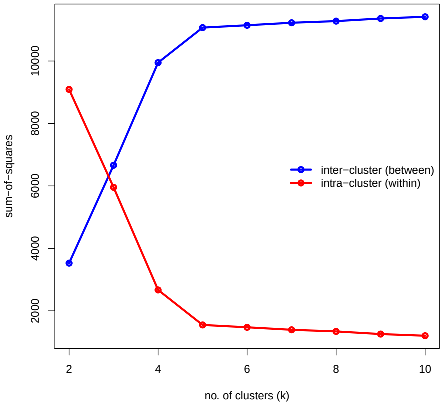
Short answer: you can't

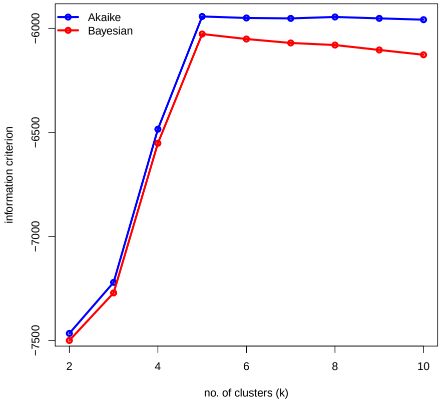
Because data is unlabelled the correct number of k is ambiguous

However we can plot some indices as a function of k to help us evaluate cluster validity:

- Within and between clusters sum-of-squares distances
- Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC) when using distribution-based methods
- Silhouette plot

How do we determine the correct number of clusters?





How do we determine the correct number of clusters?

The `NbClust` package provides 30 different cluster validity metrics. A majority vote can be taken to deduce the appropriate number of clusters

```
library(NbClust)
NbClust(data, distance, method, min.nc, max.nc, index)
# data - data frame
# distance - similarity measure e.g "euclidean"
# method - clustering algorithm e.g "kmeans"
# min.nc - min number of clusters to consider
# max.nc - max number of clusters to consider
# index - which indices to compute, "all" computes all of them
```

Note:

- Indices *only* give us a ballpark range for "correct" number of clusters
- Are they biologically relevant? Prior knowledge to the rescue
- E.g how many different phenotypes are you expecting?